Report on the Interpretation Model - DEEPLIFT

Reasons to use deeplift model for interpretations rather than other Backpropagation methods which uses gradients :

It deals with the following two problems which come across while using the other techniques:

1. **Saturation problem :** When value of the output function becomes constant or it saturates i.e gradient becomes zero, after a certain input value, the importance of the input features lying after that saturation point also becomes zero which is a misleading result.

However, the contribution scores calculated buy the DEEPLIFT method can be non zero even if gradient at a point is zero.

2. **Thresholding problem :** Sometimes there is a sudden change in the gradients such as in a following linear graph, which gives contribution scores with larger difference even in the infinitesimally small change in the input.

   However, DEEPLIFT method doesn't arises such problems due to the concept of difference-from-reference. It gives continuous values in such regions.

Note: *For a better understanding on the other available methods. Please refer PPT-1. And for a complete study of deeplift model and its advantages, refer PPT-2.*

Note: First clone the deeplift repository from this link .

Brief explanation of the DEEPLIFT functions used for testing:

Input to the model: Pretrained CNN model + dataset used for the CNN + reference input image

Output of the model: Computed importance scores for each pixel of an input image.

1. Given a saved model weights file with extension '.h5', ***convert_model_from_saved_files*** function create a revealcancel model (or say deeplift model using revealcancel method explained in the deeplift paper or ppts).

2. ***compile_func*** function compiles the generated deeplift model.

3. ***deeplift.util.run_function_in_batches*** function can be used for a sanity check that the converted model (deeplift model) is exactly same in behaviour with the original model.

4. Set the desired layers from the loaded model as the input and output layer, for which we want to calculate the importance scores using ***get_target_contribs_func*** function.

5. Finally, calculate the importance scores of the selected input layer for the selected output layer using ***score_func*** function.

Method used for choosing the correct reference image :

1. Choose one for which output of the classification is almost same for all classes. Like, for binary classification, output probability is around 0.5 for both the classes.

One way to look for those images is to search over the whole training dataset predictions and choose the one lying in the required proximity. Or one can try to design such images or download from the internet which you think can be suitable for the dataset.

2. In order to get better results, we can use many such reference images and combine their outputs.

Note: *There is no such proper function created in the code, but can be added in future.*

Types of concepts one can extract from the model prediction results using DEEPLIFT:

1. Important features / pixels present in an image responsible for its corresponding prediction result:

This is done using a threshold value for the computed scores we get after applying deeplift model over an image.

2. Comparable Features which resulted the prediction in one class and not the another i.e. the features that can be used for disguistion between two classes.

This is done by subtracting the obtained scores value corresponding to one class from another class for which we want to get distinguishable features (one can obtain the score for a particular class by specifying the task_id which is the class value as 0, 1, 2..etc), for each pixels of an image. And then apply the threshold as above to get the important or top contributing pixels.

3. Essence of a class i.e. the features which are unique to a particular class:

This is done by counting the number of times a feature is being resulted as an important one for the images which are correctly predicted and belongs to the class for which we want to get the essence of. After getting count for each feature and for each class, we can obtain the essence or features important to a class by getting the features with the highest count.

Note: *In order to get the features which are unique to a particular class, we must increase the count only when a feature is resulted important for the desired class and not for any other classes at the same time.*

Note: *Above ways of extraction depends upon the type of dataset one have. The way mentioned above is for the graph dataset when we have subgraphs as the features. The way can very for the dataset like MNIST, cats and dogs, etc., those are implemented in the attached jupyter notebooks.*

<u>Description about the attached files:</u>

1. histograms.ipynb: subgraph information is being stored using the lines of code inside this file.
2. ptc_model.ipynb: this stores the cnn model after training into a '.h5' file for ptc dataset.
3. ptc_model_test.ipynb: this involves the deeplift part and the final results for ptc dataset.
4. mnnist_model.ipynb, mnnist_model_test.ipynb: same as above for Mnnist dataset
5. cats_and_dogs.ipynb, cats_and_dogs_test.ipynb: for cats and dogs dataset